LANDSLIDE: SYSTEMATIC DYNAMIC RACE DETECTION IN KERNEL-SPACE

Ben Blum, Jiri Simsa, Garth Gibson (Carnegie Mellon University)

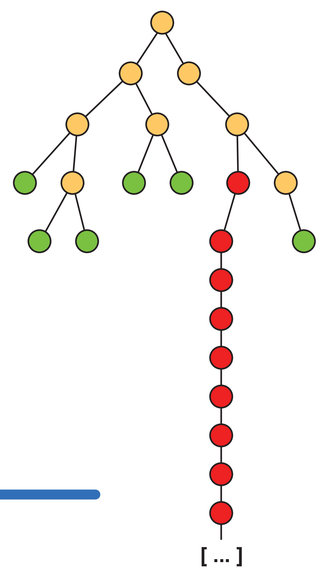
MOTIVATION & APPROACH

- Concurrency testing in kernel is harder than in user-space.
- Long-running stress tests are de facto testing method
  - Exposes race conditions at random
    - "If a preemption occurs at just the right time..."
  - Cryptic panic messages or machine reboots
  - Attempting to exercise as many internal states as possible
- Assumptions that describe user-space do not hold in the kernel
- Can instead leverage common kernel abstractions
  - Kernel stacks, runqueues, dynamic memory allocator
  - Use these to make educated guesses for when to preempt
  - Provide helpful debugging information upon finding bugs

```
[SCHEDULE]      thread 4 vanished
[SCHEDULE]      switched threads 4 -> 3
[MEMORY]        USE AFTER FREE - read from 0x0015a8f0 at eip 0x00104209
[MEMORY]        Heap contents: {...}
[MEMORY]        [0x15a8f0 | 4136] was allocated by TID3 at (...)
[MEMORY]                          and freed by TID4 at (...)
[BUG!]          ****     A bug was found!    ****
[BUG!]          **** Decision trace follows. ****
[BUG!]          1:  1347079 instructions, old 3 new 4, current 4
[BUG!]              TID3 at 0x00105a10 in context_switch,
[BUG!]                      0x001041f4 in thread_fork,
[BUG!]                      0x0010362b in thread_fork_wrapper
[BUG!]          2:  1350725 instructions, old 4 new 3, current 3
[BUG!]              TID3 at 0x00105a10 in context_switch,
[BUG!]                      0x00104681 in yield,
[BUG!]                      0x00104570 in vanish,
[BUG!]                      0x00103708 in vanish_wrapper
[BUG!]          Stack: TID3 at 0x00104209 in thread_fork,
[BUG!]                      0x0010362b in thread_fork_wrapper
[BUG!]          Total decision points 24, total backtracks 5
[BUG!]          Average instrs/decision 16155, average branch depth 5
```

IDENTIFYING BUGS

- False-negative-oriented approach
- Definite bug-detection techniques
  - Deadlock
  - Use-after-free, double-free, etc
  - Kernel panics, assertion failures
- Probable bug-detection techniques
  - Infinite loops and livelock
    - Use prior structure of execution tree to judge progress



FUTURE WORK

EDUCATION

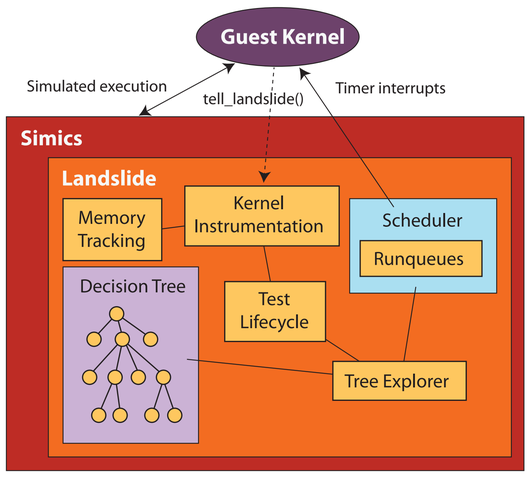
- Use as a teaching tool
  - Pebbles (CMU 15-410)
    - Change the specification to ease implementing kernel model
  - Pintos (Stanford)
- Use as a grading tool
  - TAs of 15-410 could use Landslide to augment their bug-finding ability

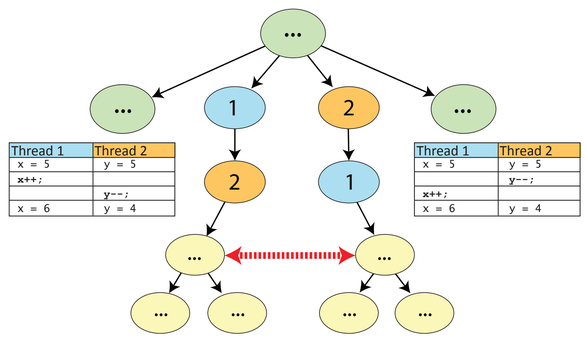
LINUX

- Virtualisation
  - Compared to Simics, lose a lot of information
- Modelling device drivers
  - Device interrupts as a new source of nondeterminism
- SMP: Enhance concurrency model to test multicore execution

LANDSLIDE DESIGN

- Controls system nondeterminism using timer interrupts
- When test case ends, rewinds machine state to a past "decision point" and force a different thread to run
- Uses dynamic partial order reduction to prune state space





CASE STUDY: THE PEBBLES KERNEL

- 15-410: Operating System Design and Implementation
- Project 3: students write a kernel in 6 weeks
- "Pebbles" is a minimal UNIX-like system call specification

USER STUDY

- Met with five groups of students in spring 2012.
- Of those, four groups invested enough time to get Landslide working
  - 100 minutes of instrumentation time on average
- All groups found bugs using Landslide; two found race conditions

RELATED WORK

- Systematic testing
  - MaceMC: Killian et al. NSDI 2007
    - Liveness properties (annotation), random walking
  - CHESS: Musuvathi et al. PLDI 2007
    - Iterative context bounding: search with fewer preemptions first
  - MoDist: Yang et al. NSDI 2009
    - Network and disk model checking for distributed systems
  - dBug: Simsa et al. SSV 2010, RV 2012
    - Partial order reduction, libc interposition
  - SimTester: Yu et al. VEE 2012
    - Simics, single interrupt injection, focus on device drivers
- Data race detection
  - Eraser: Savage et al. ACM TOCS 1997
    - Data race detection with lock-set tracking and annotations
  - DataCollider: Erickson et al. OSDI 2010
    - Random memory access sampling, targetted at kernel code
  - RacePro: Laadan et al. SOSP 2011
    - Inter-process races with system calls as points of interest

Intel Science & Technology Center for Cloud Computing

Carnegie Mellon University · Georgia Tech · intel · PRINCETON UNIVERSITY · UC Berkeley